Relocator

COPYRIGHT © 1979 by
Technical Systems Consultants, Inc.
P.O. Box 2570
West Lafayette, Indiana 47906
All Rights Reserved

DISCLAIMER

The supplied software is intended for use only as described in the manual. Use of undocumented features or parameters may cause unpredictable results for which Technical Systems Consultants, Inc. cannot assume responsibility. Although every effort has been made to make the supplied software and its documentation as accurate and functional as possible, Technical Systems Consultants, Inc. will not assume responsibility for any damages incurred or generated by such material. Technical Systems Consultants, Inc. reserves the right to make changes in such material at any time without notice.

COPYRIGHT INFORMATION

This entire manual is provided for the personal use and enjoyment of the purchaser. Its contents are copyrighted by Technical Systems Consultants, Inc., and reproduction, in whole or in part, by any means is prohibited. Use of this program, or any part thereof, for any purpose other than single end use by the purchaser is prohibited.

THE TSC 6800 RELOCATOR

SL68-28

Copyright (C) 1977 by
Technical Systems Consultants, Inc.
Box 2574; W. Lafayette, IN 47906

The TSC 6800 RELOCATOR is a very useful tool for any system owner, especially those who do assembly language programming. It can move blocks of information from one location in RAM to another. It can also relocate machine code programs from one place in RAM to another or from tape into RAM. Many variations are possible as you will see from using the RELOCATOR. The program is very easy to use as it prompts the user for all the information it needs. This manual explains the prompts and what they require in response. Included are 2 example relocations and the information necessary to relocate other TSC software. Also included is a co-resident link for the TSC TEXT EDITING SYSTEM and the TSC MNEMONIC ASSEMBLER.

First of all, here are some hints on how to respond to computer prompts. When asked a question by the computer, in general type a 'Y' for yes or an 'N' for no. When entering addresses, it is not necessary to type leading zeros. A return must be typed to terminate the address. Note that only the last 4 digits typed are accepted. Thus if you detect an error in typing before hitting a return, you can type a few zeros and then type the correct address.

WHAT THE PROMPTS MEAN:

1. PRESENT PROGRAM; BEGIN ADDRESS? ... Hexadecimal address of the first byte to be relocated.

2. PRESENT PROGRAM; END ADDRESS? ... Hexadecimal address of the last byte to be relocated.

3. MOVE TO? ... Hexadecimal address of the location to which you are moving the present program.

4. FIX REFERENCES? ... Typing an 'N' will cause the program delimited in steps 1 and 2 above to be moved exactly as is, byte for byte, to the location specified in step 3 above. You will then exit the RELOCATOR. A 'Y' will allow the program to be relocated, fixing any extended addressing references that require a change. For a further description, read the section titled 'FIXING REFERENCES'.

5. LOAD FROM TAPE? ... Type an 'N' if the program you are relocating is in RAM. If you are relocating directly from tape, type a 'Y'. The program will go into a load mode, much as if you had typed an 'L' into a MIKBUG monitor. If there is an error during loading, you will be prompted 'LOAD ERROR! TRY AGAIN?'. Typing a 'Y' at this point will put you back into the load mode. An 'N' will cause an exit from the relocator program. If there are no load errors, upon receiving an 'S9' the computer will report, '... LOAD COMPLETED.' At this point the RELOCATOR will pause until you type a space.

6. DATA BLOCKS? ... If the program you are relocating is made of executable code only, type an 'N'. If there are blocks of data, print strings or etc., type a 'Y'. You will then be asked for 'BEGIN ADDRESS' and 'END ADDRESS' for as many blocks as you wish to enter. THESE BLOCKS MUST BE ENTERED IN ORDER. That is, the block with the lowest starting address must come first, the second lowest starting address comes next, and so on. To end the entering process, enter an 'FFFF' as the begin address of a block. For a more complete description of what is considered a 'data block,' see the section titled 'DATA BLOCKS'.
   NOTE: These addresses are placed on a stack beginning at the end of the RELOCATOR ($06AD). Be sure you have enough RAM there for the number of blocks you enter!

7. ALTER RANGE? ... The 'range' is initially set to the beginning and the end of the program you are moving. This means that any JMP, JSR or other extended address instruction within that range will have an offset added to its reference (2nd and 3rd bytes) when relocated. Any extended instruction outside the range will be moved exactly as is, thus allowing monitor calls, external routine calls, etc. to be properly relocated. If you want the range left as is, type an 'N'. If you wish to change it, type a 'Y'. You will then be asked for the beginning and ending addresses of the new range.

8. FIX FDB'S? ... An FDB is a pseudo-op in standard 6800 assemblers. Its function is to define 2 bytes of RAM to be some specific value. If there are no FDB's in your program, type an 'N' and you are finished. If there are FDB's which are used to set up constants or data as opposed to addresses, they should not be changed, so type an 'N' in this case also. A common use for FDB's is to setup addresses for jump tables, command tables, etc. If your program has FDB's which setup addresses, but all those addresses are outside the range used for relocation, type an 'N' as they should not be altered. If the addresses are within the

range, you have two choices:

(1) If the locations of the FDB's themselves are within range, type a 'Y'.
(2) If the locations of the FDB's themselves are outside the range, type an 'O' or any other character.

Now you will be asked for the address of the FDB itself. You may enter as many FDB's as you like and they need not be in any order. When finished, type an 'FFFF' as an address to stop the input mode.

9. RELOCATION COMPLETED!! ... Self-explanatory!

LOADING FROM TAPE:

The TSC 6800 RELOCATOR has its own tape load routine which will read Motorola Mikbug format tapes. As each record is read, an offset is added to the record's loading address if that address is within the range of the program. When the LOAD subroutine is called, the first thing it does is turn on the reader control bit of the Mikbug control PIA. Next a string of control characters is sent out. The string is called TAPEON (at location $05AF) and is presently set to four nulls. If your tape system requires some special control characters, you may patch them into this string. DO NOT remove the '4' at the end, however. Similarly when the tape is completely loaded or when an error is received, the reader control bit of the Mikbug control PIA is turned off. Then a string called TAPOFF (at location $05B4) is sent out. TAPOFF is currently set to four nulls, but you may replace them with any control characters your tape system requires. Again, DO NOT remove the '4' at the end of the string.

Note that the LOAD routine is written as a subroutine and may therefore be called from another program if desired. Before calling it, however, you must setup an appropriate OLDPTR ($0213), OBJEND ($0217), OFFSTL ($021D), and OFFSTR ($021E). These would generally be set to $0000, $FFFF, $00 and $00 for a normal load operation.

FIXING REFERENCES:

It is not usually possible to directly move a program from one location in memory to another due to the extended references (the full 2 byte addresses in 3 byte instructions). For example, if you have an instruction which says JUMP to $1012, when the program is moved up by hex one hundred bytes it cannot stay the

same but must be changed to JUMP to $1112. This is what is meant by 'fixing references.' The TSC 6800 RELOCATOR searches thru the instructions as they are moved and any extended references are singled out. These extended addresses are then compared to the range begin and end and if inside the range, the proper offset is added to the address. Any references to outside the range are left unchanged so that jumps to external routines, calls to monitor routines, etc. will be properly relocated.

DATA BLOCKS:

Almost every program has areas which contain some type of data as opposed to executable code. When relocating, we must know where these blocks are for if we did not, there would be no way of knowing what was data and what was instructions. The instructions must have their extended references (the 2nd and 3rd bytes of 3 byte instructions) adjusted. The data must be transferred as is, byte for byte. This is the reason for specifying DATA BLOCKS. All bytes within the begin and end addresses specified (inclusive) will be relocated exactly as they are.

How do you know what should be specified as a 'DATA BLOCK'? If you have a source listing, it is generally quite easy. Any code generated by an RMB, FCB, FCC or FDB should be considered data. That's usually all there is to it! Of course if you wanted, instructions could be placed in a DATA BLOCK which would cause them to be relocated as is without fixing their references. Sometimes you may need to directly move a 3 byte immediate instruction. See the section titled '3 BYTE IMMEDIATE INSTRUCTIONS' for further details.

If you don't have a source listing, finding the data blocks becomes more of a problem. One solution is to put the object code thru a disassembler and then search out all data. Study the example relocations included for more insight.

3 BYTE IMMEDIATE INSTRUCTIONS:

There is one type of instruction which can cause problems for a relocator. That being a 3 byte immediate instruction of which there are three in the 6800 microprocessor:

LOAD INDEX REGISTER IMMEDIATE ($CE)
LOAD STACK POINTER IMMEDIATE ($8E)
COMPARE INDEX REGISTER IMMEDIATE ($8C)

In most cases the immediate bytes are an address. If that address is in range, it will be offset, otherwise it will be

directly relocated. This is the way it should be in almost all instances. You must keep an eye on the LDS command, however. Often one will say LDS immediate with an address outside the program, because you are setting up an external stack. Thus the stack will remain in the same place even though the program has been moved. If the stack is still out of the way, you have no problem, but it is something to look out for.

Another problem is in loading data into the index register or comparing the index register to data as opposed to an address. If the data is a number which is lower than the value of RANGE BEGIN or higher than the value of RANGE END, you have no problem. If, however, the data is a number inside the range, it will be altered as it looks like an address to the RELOCATOR. Although this does not occur often, it will give you an incorrect relocation.

To prevent these problems, you must know whether each occurence of a 3 byte immediate instruction contains immediate data or an immediate address. If it is an address, there will likely be no problem. If it is data, you should setup all 3 bytes of the instruction as a 'DATA BLOCK' as described above.

ADAPTING TO YOUR SYSTEM:

Adapting to your particular system is a very simple task. You must supply two routines. One is an output routine which outputs the A accumulator to your display and returns without affecting any other registers. The second is an input routine which inputs a character from your keyboard into the A accumulator and returns without affecting any other registers. You must patch the addresses of these routines into the RELOCATOR at $0220 and $0223. Upon completing a relocation, the program will jump to the address stored at MONITR ($0226). You may patch any address you like here, such as the re-entry point of your monitor. If you are using a MIKBUG monitor, these 3 addresses are already set and need not be altered.

You may need to alter the location of the stack which is presently setup at $0FFF. If this location is inconvenient for your particular system, you may change it by patching in the desired address at $0201 in the RELOCATOR.

See the section titled 'LOADING FROM TAPE' for instructions on adapting to your particular tape system.

SAMPLE RELOCATIONS:

1) The TSC 6800 Relocator

This sample will relocate the TSC 6800 RELOCATOR itself. The program starts at $0200 and ends at **$06B3** Let's assume we want to move it to $3200. Here is a copy of what the prompts and responses look like:

```
* TSC 6800 RELOCATOR *
PRESENT PROGRAM.
BEGIN ADDRESS? 200
  END ADDRESS? 6B3
      MOVE TO? 3200
FIX REFERENCES? Y
LOAD FROM TAPE? N
DATA BLOCKS? Y

BEGIN ADDRESS? 206
  END ADDRESS? 21E

BEGIN ADDRESS? 5B1
  END ADDRESS? 6B3

BEGIN ADDRESS? FFFF
ALTER RANGE? N
FIX FDB'S? N


RELOCATION COMPLETED !!!
```

Note that the stack remains at $0FFF. It may be necessary to change its location. The stack is set immediately upon entering the program. Thus in the relocated version, it is set at $3200. The instruction there is an LDS #. Change the $0FFF there if necessary.

2) TSC Space Voyage

SPACE VOYAGE actually begins at $0000 and ends at $0FFE. The first part is temporary storage in page 0, however, and cannot be moved. We will relocate only the program beginning at $0100. Let's assume we want the program moved to $1000. This relocation has an example of FDB's. They make up a jump table and thus are addresses. The FDB's themselves are outside the range, however, so an 'O' must be typed in response to the prompt, 'FIX FDB'S?' Here is what the relocation of SPACE VOYAGE looks like:

```
* TSC 6800 RELOCATOR *
PRESENT PROGRAM:
BEGIN ADDRESS? 100
  END ADDRESS? FFE
      MOVE TO? 1000
FIX REFERENCES? Y
LOAD FROM TAPE? N
DATA BLOCKS? Y

BEGIN ADDRESS? C55
  END ADDRESS? FFE

BEGIN ADDRESS? FFFF
ALTER RANGE? N
FIX FDB'S? O
ADDRESS? D5
ADDRESS? D7
ADDRESS? D9
ADDRESS? DB
ADDRESS? DD
ADDRESS? DF
ADDRESS? E1
ADDRESS? E3
ADDRESS? E5
ADDRESS? E7
ADDRESS? FFFF


RELOCATION COMPLETED !!!
```

INFORMATION FOR RELOCATING OTHER TSC PRODUCTS:

1) Micro BASIC Plus

```
BEGIN ADDRESS: 100
  END ADDRESS: D4F

    DATA BLOCKS? Y
    BEG ADDR: 10F
    END ADDR: 18A
              [301
              [307
              [498
              [4A5
              [B36
              [B43
              [D11
              [D29
              [D4D
              [D4F

    FIX FDB'S? Y
    ADDR: 114
          119
          11E       169
          123       16E
          128       173
          12D       178
          132       17E
          137       183
          13C       188
          141       B36
          146       B38
          14B       B3A
          150       B3C
          155       B3E
          15A       B40
          164       B42
```

THINGS TO CHECK:

a) The end of memory is called MEMEND in the program and is setup at $010F (before relocation). Change this address to suit your system and new program location.

b) Another thing to look out for is the external routine jump at $0701. Change the 1F00 there to the address of your external routine. If you have no external routine, you must point this subroutine jump to an RTS command. For instance, you could change the 1F00 to the address of TSTLE2 ($075E before relocation).

2) Stack Oriented Arithmetic Processor (SOAP)

```
BEGIN ADDRESS: 100
  END ADDRESS: 64C

    DATA BLOCKS? Y
    BEG ADDR: 100
    END ADDR: 11C

    FIX FDB'S? N
```

3) Disassembler

```
BEGIN ADDRESS: 1900
  END ADDRESS: 1F14

    DATA BLOCKS? Y
    BEG ADDR: 197A
    END ADDR: 199A
             ┌1A94
             └1AB3
             ┌1BF9
             └1F14

FIX FDB'S? Y
ADDRESS: 1A94
         1A96
         1A98
         1A9A
         1A9C
         1A9E
         1AA0
         1AA2
         1AA4
         1AA6
         1AA8
         1AAA
         1AAC
         1AAE
         1AB0
         1AB2
```

MAKING THE TSC EDITOR AND ASSEMBLER CO-RESIDENT:

Following is a description of the steps necessary to relocate the TSC TEXT EDITING SYSTEM, allowing co-resident operation with the TSC 6800 MNEMONIC ASSEMBLER.

1) Load the RELOCATOR
2) Move it to $3200 and set its stack pointer to $3FFF (location $3201 after relocation must be changed to $3F)
3) Load the TSC TEXT EDITING SYSTEM
4) Load the program called 'LAS' which has been included with this documentation. Type in all code generated by that program
5) Relocate the Editor-LAS pair according to the instructions given below. (Begin execution of the RELOCATOR at $3200)
6) Load the TSC 6800 MNEMONIC ASSEMBLER
7) Begin execution at $1700. See the LAS program for instructions on use and on adapting to a system larger than 16K.

RELOCATING THE EDITOR-LAS PAIR:

```
BEGIN ADDRESS: 0200
  END ADDRESS: 1559
      MOVE TO: 1700

DATA BLOCKS? Y
BEG ADDR: 0212
END ADDR: 0354
          [044C
          [044D
          [0458     [1491
          [045E     [14D2
          [0464     [150D
          [0470     [1512
          [0476     [1525
          [0482     [1532
          [0946     [1558
          [0955     [1559
          [0982
          [0988
          [0A31
          [0A47
          [0BF2
          [0C07
          [0C77
          [0C86
          [0D7F
          [0DCA
          [0FCA
          [0FD3
          [10B4
          [10CF
          [1241
          [125B
```

```
ALTER RANGE? Y
BEGIN ADDRESS: 01FF
  END ADDRESS: 1559

FIX FDB'S? Y
ADDRESS:  021B      02CE
          021F      02D2
          0228      02D9
          022C      02E4
          0235      02EA
          023C      02F4
          0241      02F8
          0245      02FF
          024E      0305
          0252      030C
          025B      0310
          0261      0316
          0268      031C
          026C      0320
          0272      0329
          027B      032D
          027F      0335
          0288      0339
          028C      033D
          0292      0344
          0299      0348
          029E      0949
          02A5      094F
          02AF      0953
          02B4      1245
          02B8      124C
          02C2      1252
          02C6      1259
```

```
* TSC EDITOR-ASSEMBLER CORESIDENT LINK
*
* COPYRIGHT (C) 1977 BY
* TECHNICAL SYSTEMS CONSULTANTS, INC.
* P.O. BOX 2574; W. LAFAYETTE, IN 47906
* (317) 742-7509
*
* THE PURPOSE OF THIS PROGRAM IS TO SETUP THE NECESSARY
* POINTERS IN THE TSC ASSEMBLER AND TO SAVE CERTAIN
* POINTERS OF THE TSC EDITOR TO ALLOW THEM TO RUN CO-
* RESIDENT.  WHEN IN THE EDITOR AND READY TO ASSEMBLE
* YOUR FILE, TYPE 'LAS' FOR 'LINK ASSEMBLER'.  YOU WILL
* BE ASKED 'LISTING OR TAPE?'.  AN 'L' WILL GIVE YOU A
* LISTING WHILE A 'T' WILL PRODUCE A MIKBUG FORMAT TAPE.
* WHEN THE ASSEMBLY IS COMPLETE, CONTROL WILL RETURN TO
* THE EDITOR.  USING LAS DELETES THE LOG COMMAND FROM THE
* EDITOR.  YOU MAY STILL USE STOP TO EXIT.  THIS LINK
* PROGRAM ASSUMES THE 'MEM' OPTION OF THE ASSEMBLER WILL
* NOT BE USED.  IF YOU DO WISH TO USE IT, EXIT THE PROGRAM
* WHEN ASKED 'LISTING OR TAPE?' AND SET UP THE MEM OPTION
* POINTER.  THEN RESUME EXECUTION AT $150F IN THE LINK
* (OR $2A0F AFTER RELOCATION).  THIS PROGRAM IS SETUP FOR
* A 16K SYSTEM.  IF YOU HAVE A LARGER SYSTEM, CHANGE
* MEMEND IN THE EDITOR ($0212 BEFORE RELOCATION; $1712
* AFTER) AND SETUP AN ADEQUATE SYMBOL TABLE BY CHANGING
* THE TABLE BEGIN POINTER AT $14BD (OR $29BD AFTER RELO-
* CATION) AND THE TABLE END POINTER AT $14BF (OR $29BF
* AFTER RELOCATION).  BE SURE THE NUMBER OF BYTES IN THE
* TABLE IS A MULTIPLE OF 8.
*
              * CHANGES TO EDITOR

0212                      ORG    $0212
0212 3A FF                FDB    $3AFF      SET MEMORY END

028E                      ORG    $028E
028E 4C                   FCC    'LAS'      PUT LAS IN COMMAND TABLE
028F 41 53
0291 00                   FCB    0
0292 14 D3                FDB    LAS

0358                      ORG    $0358
0358 CE 15 59             LDX    #BEGPT2    SETUP NEW BEGIN PTR

              * EXTERNAL EQUATES

1491                      ORG    $1491
0326            P1INIT    EQU    $0326
03B1            PASONE    EQU    $03B1
036F            P2INIT    EQU    $036F
03D9            PASTWO    EQU    $03D9
036F            P3INIT    EQU    $036F
```

```
05BB                PASTHR  EQU     $05BB
0040                LBLBEG  EQU     $0040
0042                LBLEND  EQU     $0042
0044                SRCBEG  EQU     $0044
0046                SRCEND  EQU     $0046
0048                LINBYT  EQU     $0048
0097                FILBEG  EQU     $0097
0099                FILEND  EQU     $0099
005E                ZONE1   EQU     $005E
0060                ZONE2   EQU     $0060
006A                NUMFLG  EQU     $006A
008F                INZFLG  EQU     $008F
00BB                BUFFER  EQU     $00BB
0058                SPCPT1  EQU     $0058
0096                HEDCNT  EQU     $0096
0040                TEMP    EQU     $0040
0C62                MAKSP5  EQU     $0C62
0203                RESTRT  EQU     $0203
0206                INCH    EQU     $0206
0483                PSTRNG  EQU     $0483

                    * TEMPORARY STORAGE

1491                ZONE1X  RMB     2
1493                ZONE2X  RMB     2
1495                NMFG2   RMB     2
1497                TMPEND  RMB     37
14BC                TMPBEG  RMB     1
14BD 3B 00          LBLBG2  FDB     $3B00       LABEL TABLE BEGIN ADDR.
14BF 3F FF          LBLED2  FDB     $3FFF       LABEL TABLE END ADDR.
14C1 4C             LSTORT  FCC     'LISTING OR TAPE? '
14C2 49 53
14C4 54 49
14C6 4E 47
14C8 20 4F
14CA 52 20
14CC 54 41
14CE 50 45
14D0 3F 20
14D2 04                     FCB     4

                    * ENTRY POINT UPON EXITING THE EDITOR

14D3 DE 97          LAS     LDX     FILBEG      GET FILE BEGIN
14D5 9C 99                  CPX     FILEND      ANY SOURCE IN FILE?
14D7 27 7C                  BEQ     RSTART      IF NOT, BACK TO EDITOR
14D9 DF 44                  STX     SRCBEG      SET SOURCE BEGIN
14DB DE 99                  LDX     FILEND      SET SOURCE END
14DD 09                     DEX
14DE DF 46                  STX     SRCEND
14E0 86 03                  LDA A   #3          SET LINE BYTE COUNT
14E2 97 48                  STA A   LINBYT
14E4 CE 00 BB               LDX     #BUFFER     SAVE EDITOR DATA
14E7 DF 58                  STX     SPCPT1
```

```
14E9 CE 00 96           LDX    #REDCNT
14EC DF 40              STX    TEMP
14EE CE 14 BC           LDX    #TMPBEG
14F1 BD 0C 62           JSR    MAKSP5
14F4 DE 5E              LDX    ZONE1       SAVE ZONE1
14F6 FF 14 91           STX    ZONE1X
14F9 DE 60              LDX    ZONE2       SAVE ZONE2
14FB FF 14 93           STX    ZONE2X
14FE DE 6A              LDX    NUMFLG      SAVE NUMBER & VERIFY
1500 FF 14 95           STX    NMFG2
1503 FE 14 BD           LDX    LBLBG2      SET LABEL TABLE BEGIN
1506 DF 40              STX    LBLBEG
1508 FE 14 BF           LDX    LBLED2      SET LABEL TABLE END
150B DF 42              STX    LBLEND
150D BD 03 26           JSR    P1INIT      DO PASS 1 INITIALIZE
1510 BD 03 B1           JSR    PASONE      DO PASS 1
1513 CE 14 C1           LDX    #LSTORT     ASK 'LISTING OR TAPE?'
1516 BD 04 83           JSR    PSTRNG
1519 FE 14 BF           LDX    LBLED2      RESET LABEL END
151C DF 42              STX    LBLEND
151E BD 02 06           JSR    INCH        GET RESPONSE
1521 81 54              CMP A  #'T
1523 27 08              BEQ    TAPE
1525 BD 03 6F           JSR    P2INIT      IF LISTING, DO PASS 2
1528 BD 03 D9           JSR    PASTWO
152B 20 06              BRA    EDITOR
152D BD 03 6F  TAPE     JSR    P3INIT      IF TAPE, DO PASS 3
1530 BD 05 BB           JSR    PASTHR

                * REENTRY POINT ON EXIT FROM ASSEMBLER

1533 4F         EDITOR  CLR A               CLEAR FLAG
1534 97 8F              STA A  INZFLG
1536 CE 14 BC           LDX    #TMPBEG      RESTORE EDITOR DATA
1539 DF 58              STX    SPCPT1
153B CE 14 97           LDX    #TMPEND
153E DF 40              STX    TEMP
1540 CE 00 BB           LDX    #BUFFER
1543 BD 0C 62           JSR    MAKSP5
1546 FE 14 91           LDX    ZONE1X       RESTORE ZONE1
1549 DF 5E              STX    ZONE1
154B FE 14 93           LDX    ZONE2X       RESTORE ZONE2
154E DF 60              STX    ZONE2
1550 FE 14 95           LDX    NMFG2        RESTORE NUMBER & VERIFY
1553 DF 6A              STX    NUMFLG
1555 7E 02 03  RSTART   JMP    RESTRT       JUMP INTO EDITOR


1558 0D                 FCB    $0D

1559           BEGPT2   EQU    *            START OF FILESPACE

                        END
      NO ERROR(S) DETECTED
```

SYMBOL TABLE:

BEGPT2	1559	BUFFER	00BB	EDITOR	1533	FILBEG	0097
FILEND	0099	HEDCNT	0096	INCH	0206	INZFLG	008F
LAS	14D3	LBLBEG	0040	LBLBG2	14BD	LBLED2	14BF
LBLEND	0042	LINBYT	0048	LSTORT	14C1	MAKSP5	0C62
NMFG2	1495	NUMFLG	006A	P1INIT	0326	P2INIT	036F
P3INIT	036F	PASONE	03B1	PASTHR	05BB	PASTWO	03D9
PSTRNG	0483	RESTRT	0203	RSTART	1555	SPCPT1	0058
SRCBEG	0044	SRCEND	0046	TAPE	152D	TEMP	0040
TMPBEG	14BC	TMPEND	1497	ZONE1	005E	ZONE1X	1491
ZONE2	0060	ZONE2X	1493				

OBJECT CODE:

```
S1 05 0212 3A FF AD
S1 09 028E 4C 41 53 00 14 D3 9F
S1 06 0358 CE 15 59 62
S1 13 14BD 3B 00 3F FF 4C 49 53 54 49 4E 47 20 4F 52 20 54 53
S1 13 14CD 41 50 45 3F 20 04 DE 97 9C 99 27 7C DF 44 DE 99 EB
S1 13 14DD 09 DF 46 86 03 97 48 CE 00 BB DF 58 CE 00 96 DF 62
S1 13 14ED 40 CE 14 BC BD 0C 62 DE 5E FF 14 91 DE 60 FF 14 B1
S1 13 14FD 93 DE 6A FF 14 95 FE 14 BD DF 40 FE 14 BF DF 42 78
S1 13 150D BD 03 26 BD 03 B1 CE 14 C1 BD 04 83 FE 14 BF DF DC
S1 13 151D 42 BD 02 06 81 54 27 08 BD 03 6F BD 03 D9 20 06 C1
S1 13 152D BD 03 6F BD 05 BB 4F 97 8F CE 14 BC DF 58 CE 14 D2
S1 13 153D 97 DF 40 CE 00 BB BD 0C 62 FE 14 91 DF 5E FE 14 3E
S1 0F 154D 93 DF 60 FE 14 95 DF 6A 7E 02 03 0D 3C
S9
```

```
                        *          TSC 6800 RELOCATOR
                        *
                        *        COPYRIGHT (C) 1977 BY
                        *   TECHNICAL SYSTEMS CONSULTANTS, INC.
                        *  P O. BOX 2574; W. LAFAYETTE, IN 47906
                        *            (317) 742-7509
                        *
0200                            ORG     $0200

                        * PROGRAM START

0200 8E 0F FF   START   LDS     #$0FFF      SETUP STACK
0203 7E 03 2F           JMP     BEGIN       START THE PROGRAM

                        * TEMPORARY STORAGE

0206            TAPE    RMB     1           LOADED FROM TAPE FLAG
0207            PLAY    RMB     1           RECORDER ON FLAG
0208            FIXREF  RMB     1           FIX REFERENCES FLAG
0209            TEMP1   RMB     2           TEMPORARY REGISTER
020B            TEMP2   RMB     2           TEMPORARY REGISTER
020D            TEMP3   RMB     2           TEMPORARY REGISTER
020F            CMPREG  RMB     2           2 BYTE COMPARE REG.
0211            DRCPTR  RMB     2           DIRECT STACK POINTER
0213            OLDPTR  RMB     2           OLD PROGRAM POINTER
0215            NEWPTR  RMB     2           NEW PROGRAM POINTER
0217            OBJEND  RMB     2           END OF OLD PROGRAM
0219            RGBEG   RMB     2           RANGE BEGIN ADDRESS
021B            RGEND   RMB     2           RANGE END ADDRESS
021D            OFFSTL  RMB     1           LEFT HALF OFFSET
021E            OFFSTR  RMB     1           RIGHT HALF OFFSET

                        * EXTERNAL ROUTINE JUMPS

021F 7E E1 D1   OUTCH   JMP     $E1D1       OUTPUT ROUTINE
0222 7E E1 AC   INCH    JMP     $E1AC       INPUT ROUTINE
0225 7E E0 E3   MONITR  JMP     $E0E3       EXIT ADDRESS

                        * PRINT STRINGS

0228 08         PNEXTS  INX
0229 8D 0B      PSTRNG  BSR     PCRLF       PRINT CR AND LF
022B A6 00      PDATA   LDA A   0,X         GET CHARACTER
022D 81 04              CMP A   #4          IS IT EOT?
022F 27 10              BEQ     RETURN
0231 8D EC              BSR     OUTCH       OUTPUT IT
0233 08                 INX
0234 20 F5              BRA     PDATA
0236 FF 02 09   PCRLF   STX     TEMP1       SAVE X REGISTER
0239 CE 05 BB           LDX     #CRLF       POINT TO STRING
023C 8D ED              BSR     PDATA       PRINT THE STRING
023E FE 02 09           LDX     TEMP1       RESTORE X REGISTER
0241 39         RETURN  RTS
```

```
                    * INPUT 1 HEX CHARACTER

0242 8D DE      IN1HEX  BSR     INCH        GET CHARACTER
0244 80 47      IN1HX1  SUB A   #$47        IS IT VALID?
0246 2A 0D              BPL     INERR
0248 8B 06              ADD A   #6
024A 2A 04              BPL     IN1HX2
024C 8B 07              ADD A   #7
024E 2A 05              BPL     INERR
0250 8B 0A      IN1HX2  ADD A   #10
0252 2B 01              BMI     INERR
0254 39                 RTS                 IF SO, RETURN
0255 31         INERR   INS                 IF NOT, ERROR
0256 31                 INS
0257 7D 02 07           TST     PLAY        IS TAPE ON?
025A 27 05              BEQ     INERR2
025C 31                 INS
025D 31                 INS
025E 7E 02 FA           JMP     ERROR       IF SO, TAPE ERROR
0261 CE 06 AA   INERR2  LDX     #WHAT       ELSE REPORT KEY ERROR
0264 8D C5              BSR     PDATA
0266 20 02              BRA     INADDR      TRY AGAIN

                    * INPUT NUMBER TO X REGISTER

0268 8D BF      PINADD  BSR     PSTRNG      PRINT STRING FIRST
026A 7F 02 09   INADDR  CLR     TEMP1       CLEAR REGISTER
026D 7F 02 0A           CLR     TEMP1+1
0270 8D B0      INADD0  BSR     INCH        GET CHARACTER
0272 81 0D              CMP A   #$0D        IS IT A RETURN?
0274 27 14              BEQ     INADD2      IF SO WE'RE DONE
0276 8D CC              BSR     IN1HX1      ELSE, CHECK FOR HEX
0278 48                 ASL A               SHIFT IT OVER
0279 48                 ASL A
027A 48                 ASL A
027B 48                 ASL A
027C C6 04              LDA B   #4
027E 48         INADD1  ASL A               AND INTO REGISTER
027F 79 02 0A           ROL     TEMP1+1
0282 79 02 09           ROL     TEMP1
0285 5A                 DEC B
0286 26 F6              BNE     INADD1
0288 20 E6              BRA     INADD0      GO GET ANOTHER
028A FE 02 09   INADD2  LDX     TEMP1
028D 39                 RTS

                    * INPUT 2 HEX DIGITS

028E 8D B2      IN2HEX  BSR     IN1HEX      GET 1ST DIGIT
0290 48                 ASL A               SHIFT IT OVER
0291 48                 ASL A
0292 48                 ASL A
0293 48                 ASL A
```

```
0294 16                  TAB             SAVE IT
0295 8D AB               BSR    IN1HEX   GET 2ND CHARACTER
0297 1B                  ABA             ADD IN FIRST
0298 16                  TAB
0299 FB 02 0B            ADD B  TEMP2    ADD TO CHECKSUM
029C F7 02 0B            STA B  TEMP2
029F 39                  RTS

                * LOAD A MIKBUG FORMAT TAPE

02A0 86 3C      LOAD     LDA A  #$3C     SETUP CONTROL PIA
02A2 B7 80 07            STA A  $8007
02A5 CE 05 B1            LDX    #TAPEON  PRINT CONTROL CHRS.
02A8 8D 81               BSR    PDATA
02AA BD 02 22   LOAD1    JSR    INCH     GET CHARACTER
02AD 81 53               CMP A  #'S      IS IT AN 'S'?
02AF 26 F9               BNE    LOAD1    LOOP IF NOT
02B1 BD 02 22            JSR    INCH     GET CHARACTER
02B4 81 39               CMP A  #'9      IS IT A '9'?
02B6 27 5D               BEQ    LOAD4    IF SO WE'RE DONE
02B8 80 31               SUB A  #'1      COMPARE TO A '1'
02BA 26 EE               BNE    LOAD1    LOOP IF NOT EQUAL
02BC B7 02 0B            STA A  TEMP2    CLEAR CHECKSUM
02BF 8D CD               BSR    IN2HEX
02C1 80 02               SUB A  #2       GET BYTE COUNT - 2
02C3 B7 02 0C            STA A  TEMP2+1  SAVE IT
02C6 8D C6               BSR    IN2HEX   GET LOAD ADDRESS
02C8 B7 02 09            STA A  TEMP1
02CB 8D C1               BSR    IN2HEX
02CD B7 02 0A            STA A  TEMP1+1
02D0 FE 02 09            LDX    TEMP1
02D3 BD 05 8A            JSR    CMPARE   COMPARE OLDPTR
02D6 22 0E               BHI    LOAD2    JUMP IF OUTSIDE RANGE
02D8 FE 02 17            LDX    OBJEND
02DB BD 05 90            JSR    CMPX     COMPARE ADDRESS & OBJEND
02DE 23 06               BLS    LOAD2    JUMP IF OUTSIDE RANGE
02E0 CE 02 0F            LDX    #CMPREG  IF WITHIN RANGE,
02E3 BD 05 A2            JSR    ADDOFF   ADD IN OFFSET
02E6 FE 02 0F   LOAD2    LDX    CMPREG   GET FINAL ADDRESS
02E9 8D A3      LOAD25   BSR    IN2HEX   GET A BYTE
02EB 7A 02 0C            DEC    TEMP2+1  DEC. BYTE COUNT
02EE 27 05               BEQ    LOAD3    EXIT IF = 0
02F0 A7 00               STA A  0,X      ELSE STORE BYTE
02F2 08                  INX
02F3 20 F4               BRA    LOAD25   LOOP UNTIL DONE
02F5 7C 02 0B   LOAD3    INC    TEMP2    IS CHECKSUM RIGHT?
02F8 27 B0               BEQ    LOAD1    IF SO, GET NEXT RECORD
02FA 8D 19      ERROR    BSR    LOAD4    ERROR...TURN OFF TAPE
02FC 8D 26               BSR    DELAY    PAUSE AWHILE
02FE CE 06 90            LDX    #ERR
0301 BD 02 29            JSR    PSTRNG   REPORT ERROR
0304 BD 02 22   TRYAG    JSR    INCH     GET RESPONSE
0307 81 59               CMP A  #'Y
0309 27 07               BEQ    LOAD35   IF YES, TRY AGAIN
```

```
030B 81 4E              CMP A   #'N
030D 26 F5              BNE     TRYAG
030F 7E 02 25           JMP     MONITR      IF NO, EXIT PROGRAM
0312 7E 02 A0  LOAD35   JMP     LOAD
0315 86 34     LOAD4    LDA A   #$34        RESET CONTROL PIA
0317 B7 80 07           STA A   $8007
031A CE 05 B6           LDX     #TAPOFF     PRINT CONTROL CHARS.
031D BD 02 2B           JSR     PDATA
0320 BD 02 36           JSR     PCRLF
0323 39                 RTS

                * DELAY ROUTINE

0324 CE FF FF  DELAY    LDX     #$FFFF
0327 09        DELAY1   DEX                 DELAY AWHILE
0328 08                 INX
0329 09                 DEX
032A 08                 INX
032B 09                 DEX
032C 26 F9              BNE     DELAY1
032E 39                 RTS

                * START OF MAIN PROGRAM

032F BD 02 36  BEGIN    JSR     PCRLF       PRINT 2 LINE FEEDS
0332 BD 02 36           JSR     PCRLF
0335 7F 02 06           CLR     TAPE        CLEAR FLAGS
0338 7F 02 08           CLR     FIXREF
033B 7F 02 07           CLR     PLAY
033E CE 06 AF           LDX     #DRBEG      SETUP DIRECT POINTER
0341 FF 02 11           STX     DRCPTR
0344 CE 05 C2           LDX     #INTRO
0347 BD 02 29           JSR     PSTRNG      PRINT INTRO MESSAGE
034A BD 02 28           JSR     PNEXTS
034D CE 05 EA           LDX     #BEGADR
0350 BD 02 68           JSR     PINADD      GET BEGIN ADDRESS
0353 FF 02 13           STX     OLDPTR
0356 FF 02 19           STX     RGBEG       SET RANGE BEGIN
0359 CE 05 FA           LDX     #ENDADR
035C BD 02 68           JSR     PINADD      GET END ADDRESS
035F FF 02 17           STX     OBJEND
0362 FF 02 1B           STX     RGEND       SET RANGE END
0365 CE 06 0A           LDX     #NEWBG
0368 BD 02 68           JSR     PINADD      GET NEW BEGIN ADDRESS
036B FF 02 15           STX     NEWPTR
036E B6 02 16           LDA A   NEWPTR+1    CALCULATE OFFSET
0371 B0 02 14           SUB A   OLDPTR+1
0374 B7 02 1E           STA A   OFFSTR
0377 B6 02 15           LDA A   NEWPTR
037A B2 02 13           SBC A   OLDPTR
037D B7 02 1D           STA A   OFFSTL
0380 CE 06 3E           LDX     #FIXRFS
0383 BD 02 29           JSR     PSTRNG      ASK TO FIX REFERENCES
0386 BD 02 22           JSR     INCH        GET RESPONSE
```

```
0389 81 4E            CMP A  #'N
038B 27 03            BEQ    LDFRTP
038D 7C 02 08         INC    FIXREF     IF YES, SET FLAG
0390 CE 06 1A LDFRTP  LDX    #TAPSTR
0393 BD 02 29         JSR    PSTRNG     LOADING FROM TAPE?
0396 BD 02 22         JSR    INCH       GET RESPONSE
0399 81 59            CMP A  #'Y
039B 27 03            BEQ    LDFRT1
039D 7E 04 26         JMP    NOTAPE     IF NOT, JUMP AHEAD
03A0 7C 02 06 LDFRT1  INC    TAPE       IF SO, SET TAPE FLAG
03A3 7C 02 07         INC    PLAY
03A6 BD 02 A0         JSR    LOAD       GO LOAD TAPE
03A9 7F 02 07         CLR    PLAY
03AC BD 03 24         JSR    DELAY      PAUSE AWHILE
03AF CE 06 2B         LDX    #LOADED
03B2 BD 02 29         JSR    PSTRNG     REPORT LOAD COMPLETE
03B5 BD 02 22 WAIT    JSR    INCH       GET A CHARACTER
03B8 81 20            CMP A  #$20
03BA 26 F9            BNE    WAIT       BUT ONLY ACCEPT A SPACE
03BC 7D 02 08         TST    FIXREF     FIXING REFERENCES?
03BF 26 03            BNE    TAPFIX
03C1 7E 02 25         JMP    MONITR     IF NOT, EXIT PROGRAM
03C4 FE 02 15 TAPFIX  LDX    NEWPTR
03C7 FF 02 13         STX    OLDPTR     IF SO, FIX OLDPTR
03CA CE 02 17         LDX    #OBJEND
03CD BD 05 A2         JSR    ADDOFF     AND OBJECT END

                * ENTER DIRECT DATA BLOCKS

03D0 CE 06 AF DRBLKS  LDX    #DRBEG
03D3 FF 02 0D         STX    TEMP3      SAVE DIRECT BEGIN
03D6 CE 06 4F         LDX    #DRCTBK
03D9 BD 02 29         JSR    PSTRNG     ANY DIRECT RELOCATES?
03DC BD 02 22         JSR    INCH
03DF 81 4E            CMP A  #'N
03E1 26 05            BNE    DRBLK1     IF SO GO GET THEM
03E3 CE FF FF         LDX    #$FFFF
03E6 20 63            BRA    DIFFRG     IF NOT, JUMP AHEAD
03E8 BD 02 36 DRBLK1  JSR    PCRLF
03EB CE 05 EA         LDX    #BEGADR
03EE BD 02 68         JSR    PINADD     GET BLOCK BEGIN
03F1 8C FF FF         CPX    #$FFFF     FINISHED?
03F4 27 55            BEQ    DIFFRG     IF SO, JUMP AHEAD
03F6 8D 0A            BSR    ENTER      PUT ADDRESS ON STACK
03F8 CE 05 FA         LDX    #ENDADR
03FB BD 02 68         JSR    PINADD     GET BLOCK END
03FE 8D 02            BSR    ENTER      PUT IT ON STACK
0400 20 E6            BRA    DRBLK1     LOOP BACK
0402 7D 02 06 ENTER   TST    TAPE       LOADED FROM TAPE?
0405 27 09            BEQ    ENTER0     IF NOT GO AHEAD
0407 CE 02 09         LDX    #TEMP1
040A BD 05 A2         JSR    ADDOFF     IF SO, ADD OFFSET
040D FE 02 09         LDX    TEMP1
0410 FF 02 0B ENTER0  STX    TEMP2      SAVE ADDRESS
```

```
0413 FE 02 0D           LDX     TEMP3       POINT TO DIRECT STACK
0416 B6 02 0B           LDA A   TEMP2       PUT ADDRESS ON STACK
0419 A7 00              STA A   0,X
041B B6 02 0C           LDA A   TEMP2+1
041E A7 01              STA A   1,X
0420 08         ENTER1  INX                 FIX DIRECT STACK PTR.
0421 08                 INX
0422 FF 02 0D           STX     TEMP3
0425 39                 RTS
0426 7D 02 08   NOTAPE  TST     FIXREF      FIXING REFERENCES?
0429 26 A5              BNE     DRBLKS      IF SO, GO ENTER DIRECTS
042B CE 00 00           LDX     #$0000      IF NOT, MAKE THE
042E FF 06 AF           STX     DRBEG       ENTIRE RAM SPACE INTO
0431 CE FF FF           LDX     #$FFFF      A DIRECT RELOCATE BLOCK
0434 FF 06 B1           STX     DRBEG+2
0437 FF 06 B3           STX     DRBEG+4
043A 20 30              BRA     LOOP        START RELOCATION

                * ROUTINE TO INCREMENT POINTERS

043C FE 02 15   INCPTR  LDX     NEWPTR
043F 08                 INX                 INCREMENT NEW POINTER
0440 FF 02 15           STX     NEWPTR
0443 FE 02 13           LDX     OLDPTR
0446 08                 INX                 INCREMENT OLD POINTER
0447 FF 02 13           STX     OLDPTR
044A 39                 RTS

                * CHANGE REFERENCE RANGE ROUTINE

044B 8D C3      DIFFRG  BSR     ENTER0      SET DIRECT STACK END
044D CE 06 5D           LDX     #CHANGE
0450 BD 02 29           JSR     PSTRNG      ASK TO CHANGE RANGE
0453 BD 02 22           JSR     INCH        GET RESPONSE
0456 81 59              CMP A   #'Y
0458 26 12              BNE     LOOP        IF NO, START RELOCATION
045A CE 05 EA           LDX     #BEGADR
045D BD 02 68           JSR     PINADD      GET RANGE BEGIN
0460 FF 02 19           STX     RGBEG
0463 CE 05 FA           LDX     #ENDADR
0466 BD 02 68           JSR     PINADD      GET RANGE END
0469 FF 02 1B           STX     RGEND

                * MAIN RELOCATION LOOP

046C FE 02 17   LOOP    LDX     OBJEND      IS OLDPTR > OBJEND?
046F BD 05 8A           JSR     CMPARE
0472 23 03              BLS     LOOP1
0474 7E 05 44           JMP     DONE        IF SO WE'RE DONE
0477 FE 02 11   LOOP1   LDX     DRCPTR      IS THIS A DIRECT BLOCK?
047A EE 00              LDX     0,X
047C BD 05 8A           JSR     CMPARE
047F 25 03              BCS     LOOP2
0481 7E 05 1E           JMP     DIRECT      IF SO, GO MOVE DIRECT
```

```
0484 A6 00      LOOP2   LDA A   0,X        MOVE OPCODE
0486 FE 02 15           LDX     NEWPTR
0489 A7 00              STA A   0,X
048B FE 02 13           LDX     OLDPTR
048E 84 30              AND A   #$30       CHECK FOR 3 BYTE INST.
0490 81 30              CMP A   #$30
0492 27 29              BEQ     MAYBE3     COULD BE 3 BYTES
0494 A6 00              LDA A   0,X
0496 81 CE              CMP A   #$CE       CHECK FOR LDX #
0498 27 29              BEQ     THREE
049A 81 8C              CMP A   #$8C       CHECK FOR CPX #
049C 27 25              BEQ     THREE
049E 81 8E              CMP A   #$8E       CHECK FOR LDS #
04A0 27 21              BEQ     THREE
04A2 81 5F              CMP A   #$5F       LOOK FOR 2 BYTE INST.
04A4 22 0B              BHI     TWO
04A6 84 F0              AND A   #$F0       LOOK FOR 1 BYTE INST.
04A8 81 20              CMP A   #$20
04AA 27 05              BEQ     TWO

                * ONE BYTE INSTRUCTION

04AC BD 04 3C   ONE     JSR     INCPTR
04AF 20 BB              BRA     LOOP       GET NEXT INSTRUCTION

                *TWO BYTE INSTRUCTION

04B1 BD 04 3C   TWO     JSR     INCPTR     POINT TO 2ND BYTE
04B4 A6 00              LDA A   0,X        MOVE IT
04B6 FE 02 15           LDX     NEWPTR
04B9 A7 00              STA A   0,X
04BB 20 EF              BRA     ONE        NEXT INSTRUCTION

04BD A6 00      MAYBE3  LDA A   0,X        CHECK 3 OR 1 BYTE INST.
04BF 85 C0              BIT A   #$C0
04C1 27 E9              BEQ     ONE

                * THREE BYTE INSTRUCTION

04C3 BD 04 3C   THREE   JSR     INCPTR     POINT TO REFERENCE
04C6 FE 02 19           LDX     RGBEG      IS IT BELOW RANGE BEG?
04C9 FF 02 0F           STX     CMPREG
04CC FE 02 13           LDX     OLDPTR
04CF EE 00              LDX     0,X
04D1 BD 05 90           JSR     CMPX
04D4 25 3C              BLO     NOFFST     IF SO, NO OFFSET
04D6 FE 02 1B           LDX     RGEND      IS IT ABOVE RANGE END?
04D9 FF 02 0F           STX     CMPREG
04DC FE 02 13           LDX     OLDPTR
04DF EE 00              LDX     0,X
04E1 BD 05 90           JSR     CMPX
04E4 22 2C              BHI     NOFFST     IF SO, NO OFFSET
04E6 FE 02 13           LDX     OLDPTR
04E9 09                 DEX
```

```
04EA A6 00              LDA A  0,X      GET OPCODE
04EC 08                 INX
04ED 81 7E              CMP A  #$7E     IS IT A JUMP?
04EF 27 0A              BEQ    OFFSET   IF SO, DO OFFSET
04F1 84 F0              AND A  #$F0     CHECK FOR PAGE 0 REF.
04F3 81 70              CMP A  #$70
04F5 26 04              BNE    OFFSET
04F7 A6 00              LDA A  0,X
04F9 27 1C              BEQ    NOFST1   IF PAGE 0, NO OFFSET
04FB A6 01      OFFSET  LDA A  1,X      ADD OFFSET TO REFERENCE
04FD BB 02 1E           ADD A  OFFSTR
0500 16                 TAB
0501 A6 00              LDA A  0,X
0503 B9 02 1D           ADC A  OFFSTL
0506 FE 02 15   NEXT    LDX    NEWPTR   STORE RESULT
0509 A7 00              STA A  0,X
050B E7 01              STA B  1,X
050D BD 04 3C           JSR    INCPTR
0510 20 9A              BRA    ONE      GET NEXT INSTRUCTION
0512 FE 02 13   NOFFST  LDX    OLDPTR   NO OFFSET ADDED
0515 A6 00              LDA A  0,X
0517 E6 01      NOFST1  LDA B  1,X
0519 20 EB              BRA    NEXT

                * MOVE DIRECT DATA BLOCK

051B BD 04 3C   DRECT0  JSR    INCPTR   BUMP POINTERS
051E A6 00      DIRECT  LDA A  0,X      MOVE ONE BYTE
0520 FE 02 15           LDX    NEWPTR
0523 A7 00              STA A  0,X
0525 FE 02 17           LDX    OBJEND
0528 BD 05 8A           JSR    CMPARE   END OF PROGRAM?
052B 27 17              BEQ    DONE     IF SO, WE'RE DONE
052D FE 02 11           LDX    DRCPTR   GET BLOCK END ADDRESS
0530 EE 02              LDX    2,X
0532 BD 05 8A           JSR    CMPARE   ARE WE THERE?
0535 26 E4              BNE    DRECT0   IF NOT, MOVE ANOTHER
0537 FE 02 11           LDX    DRCPTR   FIXUP DIRECT POINTER
053A 08                 INX
053B 08                 INX
053C 08                 INX
053D 08                 INX
053E FF 02 11           STX    DRCPTR
0541 7E 04 AC           JMP    ONE      GO TO NORMAL RELOCATION

                * CODE IS RELOCATED, CHECK FDB'S

0544 7D 02 08   DONE    TST    FIXREF   FIXING REFERENCES?
0547 27 2F              BEQ    DONE2    IF NOT, ALL DONE
0549 5F                 CLR B
054A CE 06 84           LDX    #FXFBDS
054D BD 02 29           JSR    PSTRNG   ASK TO FIX FDB'S
0550 BD 02 22           JSR    INCH     GET RESPONSE
0553 81 4E              CMP A  #'N
```

```
0555 27 21              BEQ     DONE2       IF N, ALL DONE
0557 81 59              CMP A   #'Y
0559 27 01              BEQ     DONE0       IF Y, JUMP AHEAD
055B 5C                 INC B               ELSE SET FLAG
055C 37         DONE0   PSH B               SAVE FLAG
055D CE 05 F0           LDX     #BEGADR+6
0560 BD 02 68           JSR     PINADD      GET FDB ADDRESS
0563 33                 PUL B               RESTORE FLAG
0564 8C FF FF           CPX     #$FFFF      ANY MORE FDB'S?
0567 27 0F              BEQ     DONE2       IF NOT, ALL DONE
0569 5D                 TST B               IS FDB WITHIN RANGE?
056A 26 08              BNE     DONE1       IF NOT, NO OFFSET
056C CE 02 09           LDX     #TEMP1
056F 8D 31              BSR     ADDOFF      ELSE ADD IN OFFSET
0571 FE 02 09           LDX     TEMP1
0574 8D 2C      DONE1   BSR     ADDOFF      FIXUP THE FDB
0576 20 E4              BRA     DONE0       ANY MORE?

                * ALL FINISHED ROUTINE

0578 BD 02 36   DONE2   JSR     PCRLF
057B BD 02 36           JSR     PCRLF
057E CE 06 6B           LDX     #FINE
0581 BD 02 29           JSR     PSTRNG      REPORT COMPLETION
0584 BD 02 36           JSR     PCRLF
0587 7E 02 25           JMP     MONITR      EXIT THE PROGRAM

                * TWO BYTE COMPARE ROUTINE

058A FF 02 0F   CMPARE  STX     CMPREG
058D FE 02 13           LDX     OLDPTR
0590 FF 02 09   CMPX    STX     TEMP1       COMPARE CMPREG TO TEMP1
0593 B6 02 09           LDA A   TEMP1
0596 B1 02 0F           CMP A   CMPREG
0599 26 06              BNE     CMPX1
059B B6 02 0A           LDA A   TEMP1+1
059E B1 02 10           CMP A   CMPREG+1
05A1 39         CMPX1   RTS

                * ROUTINE TO ADD IN OFFSET

05A2 A6 01      ADDOFF  LDA A   1,X         GET RIGHT HALF
05A4 BB 02 1E           ADD A   OFFSTR      ADD OFFSET RIGHT
05A7 A7 01              STA A   1,X
05A9 A6 00              LDA A   0,X         GET LEFT HALF
05AB B9 02 1D           ADC A   OFFSTL      ADD OFFSET LEFT
05AE A7 00              STA A   0,X
05B0 39                 RTS


                * STRINGS

05B1 00         TAPEON  FCB     0,0,0,0,4
05B2 00 00
```

```
05B4 00 04
05B6 00          TAPOFF   FCB     0,0,0,0,4
05B7 00 00
05B9 00 04
05BB 0D          CRLF     FCB     $D,$A,0,0,0,0,4
05BC 0A 00
05BE 00 00
05C0 00 04
05C2 2A          INTRO    FCC     '* TSC 6800 RELOCATOR *'
05C3 20 54
05C5 53 43
05C7 20 36
05C9 38 30
05CB 30 20
05CD 52 45
05CF 4C 4F
05D1 43 41
05D3 54 4F
05D5 52 20
05D7 2A
05D8 04                  FCB     4
05D9 50                  FCC     'PRESENT PROGRAM:'
05DA 52 45
05DC 53 45
05DE 4E 54
05E0 20 50
05E2 52 4F
05E4 47 52
05E6 41 4D
05E8 3A
05E9 04                  FCB     4
05EA 42          BEGADR  FCC     'BEGIN ADDRESS? '
05EB 45 47
05ED 49 4E
05EF 20 41
05F1 44 44
05F3 52 45
05F5 53 53
05F7 3F 20
05F9 04                  FCB     4
05FA 20          ENDADR  FCC     '  END ADDRESS? '
05FB 20 45
05FD 4E 44
05FF 20 41
0601 44 44
0603 52 45
0605 53 53
0607 3F 20
0609 04                  FCB     4
060A 20          NEWBG   FCC     '       MOVE TO? '
060B 20 20
060D 20 20
060F 20 4D
0611 4F 56
```

```
0613 45 20
0615 54 4F
0617 3F 20
0619 04              FCB    4
061A 4C        TAPSTR FCC    'LOAD FROM TAPE? '
061B 4F 41
061D 44 20
061F 46 52
0621 4F 4D
0623 20 54
0625 41 50
0627 45 3F
0629 20
062A 04              FCB    4
062B 2E        LOADED FCC    '...LOAD COMPLETED.'
062C 2E 2E
062E 4C 4F
0630 41 44
0632 20 43
0634 4F 4D
0636 50 4C
0638 45 54
063A 45 44
063C 2E
063D 04              FCB    4
063E 46        FIXRFS FCC    'FIX REFERENCES? '
063F 49 58
0641 20 52
0643 45 46
0645 45 52
0647 45 4E
0649 43 45
064B 53 3F
064D 20
064E 04              FCB    4
064F 44        DRCTBK FCC    'DATA BLOCKS? '
0650 41 54
0652 41 20
0654 42 4C
0656 4F 43
0658 4B 53
065A 3F 20
065C 04              FCB    4
065D 41        CHANGE FCC    'ALTER RANGE? '
065E 4C 54
0660 45 52
0662 20 52
0664 41 4E
0666 47 45
0668 3F 20
066A 04              FCB    4
066B 52        FINE   FCC    'RELOCATION COMPLETED !!!'
066C 45 4C
066E 4F 43
```

```
0670 41 54
0672 49 4F
0674 4E 20
0676 43 4F
0678 4D 50
067A 4C 45
067C 54 45
067E 44 20
0680 21 21
0682 21
0683 04                 FCB     4
0684 46         FXFBDS  FCC     'FIX FDB'
0685 49 58
0687 20 46
0689 44 42
068B 27                 FCB     $27,$53,$3F,$20,4
068C 53 3F
068E 20 04
0690 4C         ERR     FCC     'LOAD ERROR!  TRY AGAIN? '
0691 4F 41
0693 44 20
0695 45 52
0697 52 4F
0699 52 21
069B 20 20
069D 54 52
069F 59 20
06A1 41 47
06A3 41 49
06A5 4E 3F
06A7 20
06A8 07                 FCB     7,4
06A9 04
06AA 07         WHAT    FCB     7,$20,$3F,$20,4
06AB 20 3F
06AD 20 04
06AF            DRBEG   RMB     20


                        END
      NO ERROR(S) DETECTED
```

SYMBOL TABLE:

ADDOFF	05A2	BEGADR	05EA	BEGIN	032F	CHANGE	065D	CMPARE	058A
CMPREG	020F	CMPX	0590	CMPX1	05A1	CRLF	05BB	DELAY	0324
DELAY1	0327	DIFFRG	044B	DIRECT	051E	DONE	0544	DONE0	055C
DONE1	0574	DONE2	0578	DRBEG	06AF	DRBLK1	03E8	DRBLKS	03D0
DRCPTR	0211	DRCTBK	064F	DRECT0	051B	ENDADR	05FA	ENTER	0402
ENTER0	0410	ENTER1	0420	ERR	0690	ERROR	02FA	FINE	066B
FIXREF	0208	FIXRFS	063E	FXFBDS	0684	IN1HEX	0242	IN1HX1	0244
IN1HX2	0250	IN2HEX	028E	INADD0	0270	INADD1	027E	INADD2	028A
INADDR	026A	INCH	0222	INCPTR	043C	INERR	0255	INERR2	0261

INTRO	05C2	LDFRT1	03A0	LDFRTP	0390	LOAD	02A0	LOAD1	02AA
LOAD2	02E6	LOAD25	02E9	LOAD3	02F5	LOAD35	0312	LOAD4	0315
LOADED	062B	LOOP	046C	LOOP1	0477	LOOP2	0484	MAYBE3	04BD
MONITR	0225	NEWBG	060A	NEWPTR	0215	NEXT	0506	NOFFST	0512
NOFST1	0517	NOTAPE	0426	OBJEND	0217	OFFSET	04FB	OFFSTL	021D
OFFSTR	021E	OLDPTR	0213	ONE	04AC	OUTCH	021F	PCRLF	0236
PDATA	022B	PINADD	0268	PLAY	0207	PNEXTS	0228	PSTRNG	0229
RETURN	0241	RGBEG	0219	RGEND	021B	START	0200	TAPE	0206
TAPEON	05B1	TAPFIX	03C4	TAPOFF	05B6	TAPSTR	061A	TEMP1	0209
TEMP2	020B	TEMP3	020D	THREE	04C3	TRYAG	0304	TWO	04B1
WAIT	03B5	WHAT	06AA						

OBJECT CODE:

```
S1 09 0200 8E 0F FF 7E 03 2F A8
S1 13 021F 7E E1 D1 7E E1 AC 7E E0 E3 08 8D 0B A6 00 81 04 84
S1 13 022F 27 10 8D EC 08 20 F5 FF 02 09 CE 05 BB 8D ED FE DE
S1 13 023F 02 09 39 8D DE 80 47 2A 0D 8B 06 2A 04 8B 07 2A 83
S1 13 024F 05 8B 0A 2B 01 39 31 31 7D 02 07 27 05 31 31 7E A8
S1 13 025F 02 FA CE 06 AA 8D C5 20 02 8D BF 7F 02 09 7F 02 46
S1 13 026F 0A 8D B0 81 0D 27 14 8D CC 48 48 48 48 C6 04 48 E0
S1 13 027F 79 02 0A 79 02 09 5A 26 F6 20 E6 FE 02 09 39 8D 17
S1 13 028F B2 48 48 48 48 16 8D AB 1B 16 FB 02 0B F7 02 0B FE
S1 13 029F 39 86 3C B7 80 07 CE 05 B1 8D 81 BD 02 22 81 53 CB
S1 13 02AF 26 F9 BD 02 22 81 39 27 5D 80 31 26 EE B7 02 0B 74
S1 13 02BF 8D CD 80 02 B7 02 0C 8D C6 B7 02 09 8D C1 B7 02 6E
S1 13 02CF 0A FE 02 09 BD 05 8A 22 0E FE 02 17 BD 05 90 23 00
S1 13 02DF 06 CE 02 0F BD 05 A2 FE 02 0F 8D A3 7A 02 0C 27 D4
S1 13 02EF 05 A7 00 08 20 F4 7C 02 0B 27 B0 8D 19 8D 26 CE AC
S1 13 02FF 06 90 BD 02 29 BD 02 22 81 59 27 07 81 4E 26 F5 9A
S1 13 030F 7E 02 25 7E 02 A0 86 34 B7 80 07 CE 05 B6 BD 02 D5
S1 13 031F 2B BD 02 36 39 CE FF FF 09 08 09 08 09 26 F9 39 22
S1 13 032F BD 02 36 BD 02 36 7F 02 06 7F 02 08 7F 02 07 CE 6A
S1 13 033F 06 AF FF 02 11 CE 05 C2 BD 02 29 BD 02 28 CE 05 AC
S1 13 034F EA BD 02 68 FF 02 13 FF 02 19 CE 05 FA BD 02 68 67
S1 13 035F FF 02 17 FF 02 1B CE 06 0A BD 02 68 FF 02 15 B6 85
S1 13 036F 02 16 B0 02 14 B7 02 1E B6 02 15 B2 02 13 B7 02 78
S1 13 037F 1D CE 06 3E BD 02 29 BD 02 22 81 4E 27 03 7C 02 FB
S1 13 038F 08 CE 06 1A BD 02 29 BD 02 22 81 59 27 03 7E 04 15
S1 13 039F 26 7C 02 06 7C 02 07 BD 02 A0 7F 02 07 BD 03 24 50
S1 13 03AF CE 06 2B BD 02 29 BD 02 22 81 20 26 F9 7D 02 08 2B
S1 13 03BF 26 03 7E 02 25 FE 02 15 FF 02 13 CE 02 17 BD 05 8A
S1 13 03CF A2 CE 06 AF FF 02 0D CE 06 4F BD 02 29 BD 02 22 FB
S1 13 03DF 81 4E 26 05 CE FF FF 20 63 BD 02 36 CE 05 EA BD 52
S1 13 03EF 02 68 8C FF FF 27 55 8D 0A CE 05 FA BD 02 68 8D 72
S1 13 03FF 02 20 E6 7D 02 06 27 09 CE 02 09 BD 05 A2 FE 02 F0
S1 13 040F 09 FF 02 0B FE 02 0D B6 02 0B A7 00 B6 02 0C A7 E2
S1 13 041F 01 08 08 FF 02 0D 39 7D 02 08 26 A5 CE 00 00 FF 52
S1 13 042F 06 AF CE FF FF FF 06 B1 FF 06 B3 20 30 FE 02 15 65
S1 13 043F 08 FF 02 15 FE 02 13 08 FF 02 13 39 8D C3 CE 06 FF
S1 13 044F 5D BD 02 29 BD 02 22 81 59 26 12 CE 05 EA BD 02 E5
```

```
S1 13 045F 68 FF 02 19 CE 05 FA BD 02 68 FF 02 1B FE 02 17 E0
S1 13 046F BD 05 8A 23 03 7E 05 44 FE 02 11 EE 00 BD 05 8A F5
S1 13 047F 25 03 7E 05 1E A6 00 FE 02 15 A7 00 FE 02 13 84 A7
S1 13 048F 30 81 30 27 29 A6 00 81 CE 27 29 81 8C 27 25 81 09
S1 13 049F 8E 27 21 81 5F 22 0B 84 F0 81 20 27 05 BD 04 3C 28
S1 13 04AF 20 BB BD 04 3C A6 00 FE 02 15 A7 00 20 EF A6 00 4A
S1 13 04BF 85 C0 27 E9 BD 04 3C FE 02 19 FF 02 0F FE 02 13 9B
S1 13 04CF EE 00 BD 05 90 25 3C FE 02 1B FF 02 0F FE 02 13 3A
S1 13 04DF EE 00 BD 05 90 22 2C FE 02 13 09 A6 00 08 81 7E B2
S1 13 04EF 27 0A 84 F0 81 70 26 04 A6 00 27 1C A6 01 BB 02 EC
S1 13 04FF 1E 16 A6 00 B9 02 1D FE 02 15 A7 00 E7 01 BD 04 D2
S1 13 050F 3C 20 9A FE 02 13 A6 00 E6 01 20 EB BD 04 3C A6 94
S1 13 051F 00 FE 02 15 A7 00 FE 02 17 BD 05 8A 27 17 FE 02 6B
S1 13 052F 11 EE 02 BD 05 8A 26 E4 FE 02 11 08 08 08 08 FF 31
S1 13 053F 02 11 7E 04 AC 7D 02 08 27 2F 5F CE 06 84 BD 02 14
S1 13 054F 29 BD 02 22 81 4E 27 21 81 59 27 01 5C 37 CE 05 0F
S1 13 055F F0 BD 02 68 33 8C FF FF 27 0F 5D 26 08 CE 02 09 1A
S1 13 056F 8D 31 FE 02 09 8D 2C 20 E4 BD 02 36 BD 02 36 CE 3C
S1 13 057F 06 6B BD 02 29 BD 02 36 7E 02 25 FF 02 0F FE 02 65
S1 13 058F 13 FF 02 09 B6 02 09 B1 02 0F 26 06 B6 02 0A B1 19
S1 13 059F 02 10 39 A6 01 BB 02 1E A7 01 A6 00 B9 02 1D A7 AE
S1 13 05AF 00 39 00 00 00 00 04 00 00 00 00 04 0D 0A 00 00 E0
S1 13 05BF 00 00 04 2A 20 54 53 43 20 36 38 30 30 20 52 45 4B
S1 13 05CF 4C 4F 43 41 54 4F 52 20 2A 04 50 52 45 53 45 4E E9
S1 13 05DF 54 20 50 52 4F 47 52 41 4D 3A 04 42 45 47 49 4E D9
S1 13 05EF 20 41 44 44 52 45 53 53 3F 20 04 20 20 45 4E 44 58
S1 13 05FF 20 41 44 44 52 45 53 53 3F 20 04 20 20 20 20 20 BF
S1 13 060F 20 4D 4F 56 45 20 54 4F 3F 20 04 4C 4F 41 44 20 1A
S1 13 061F 46 52 4F 4D 20 54 41 50 45 3F 20 04 2E 2E 2E 4C 10
S1 13 062F 4F 41 44 20 43 4F 4D 50 4C 45 54 45 44 2E 04 46 AE
S1 13 063F 49 58 20 52 45 46 45 52 45 4E 43 45 53 3F 20 04 A1
S1 13 064F 44 41 54 41 20 42 4C 4F 43 4B 53 3F 20 04 41 4C AF
S1 13 065F 54 45 52 20 52 41 4E 47 45 3F 20 04 52 45 4C 4F 7A
S1 13 066F 43 41 54 49 4F 4E 20 43 4F 4D 50 4C 45 54 45 44 FC
S1 13 067F 20 21 21 21 04 46 49 58 20 46 44 42 27 53 3F 20 34
S1 13 068F 04 4C 4F 41 44 20 45 52 52 4F 52 21 20 20 54 52 82
S1 13 069F 59 20 41 47 41 49 4E 3F 20 07 04 07 20 3F 20 04 7A
S9
```